

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	116	coherency adj point	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/07/19 20:00
L2	252	probe adj command	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/07/19 20:01
L3	76	1 and 2	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/07/19 20:01
L4	65	3 and miss and hit	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/07/19 20:01
L5	23	4 and exclusive	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/07/19 20:02
L6	9	5 and directory	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/07/19 20:02
L7	9	6 and node	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2005/07/19 20:06

[Home](#) | [Login](#) | [Logout](#) | [Access Information](#) | [Alerts](#) |

Welcome United States Patent and Trademark Office

Search Results[BROWSE](#)[SEARCH](#)[IEEE XPLOR GUIDE](#)

Results for "((coherency<and>(probe<near/4>command)<and> miss<and>hit<and>exclusive...)"
Your search matched 0 of 1194402 documents.

[e-mail](#)

A maximum of 100 results are displayed, 25 to a page, sorted by **Relevance in Descending** order.

[» View Session History](#)[» New Search](#)**Modify Search**[» Key](#)**IEEE JNL** IEEE Journal or Magazine Check to search only within this results set**IEE JNL** IEE Journal or MagazineDisplay Format: Citation Citation & Abstract**IEEE CNF** IEEE Conference Proceeding**IEE CNF** IEE Conference Proceeding**No results were found.****IEEE STD** IEEE Standard

Please edit your search criteria and try again. Refer to the Help pages if you need assistance revisir

[Help](#) [Contact Us](#) [Privacy & ...](#)

© Copyright 2005 IEEE ...

Indexed by
Inspec



USPTO

[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)
Search: The ACM Digital Library The Guide

coherency (probe command) miss hit exclusive directory node



THE ACM DIGITAL LIBRARY

[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

Terms used

[coherency probe command miss hit exclusive directory node](#)

Found 17,989 of 157,873

Sort results by

 relevance
[Save results to a Binder](#)[Try an Advanced Search](#)

Display results

 expanded form
[Search Tips](#)[Try this search in The ACM Guide](#) [Open results in a new window](#)

Results 1 - 20 of 200

Result page: **1** [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale

1 Delayed consistency and its effects on the miss rate of parallel programs

Michel Dubois, Jin Chin Wang, Luiz A. Barroso, Kangwoo Lee, Yung-Syau Chen

August 1991 **Proceedings of the 1991 ACM/IEEE conference on Supercomputing**Full text available: [pdf\(1.01 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)**2 Query evaluation techniques for large databases**

Goetz Graefe

June 1993 **ACM Computing Surveys (CSUR)**, Volume 25 Issue 2Full text available: [pdf\(9.37 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Database management systems will continue to manage large data volumes. Thus, efficient algorithms for accessing and manipulating large sets and sequences will be required to provide acceptable performance. The advent of object-oriented and extensible database systems will not solve this problem. On the contrary, modern data models exacerbate the problem: In order to manipulate large sets of complex objects as efficiently as today's database systems manipulate simple records, query-processi ...

Keywords: complex query evaluation plans, dynamic query evaluation plans, extensible database systems, iterators, object-oriented database systems, operator model of parallelization, parallel algorithms, relational database systems, set-matching algorithms, sort-hash duality

3 Performance evaluation of the Orca shared-object system

Henri E. Bal, Raoul Bhoedjang, Rutger Hofman, Ceriel Jacobs, Koen Langendoen, Tim Rühl, M. Frans Kaashoek

February 1998 **ACM Transactions on Computer Systems (TOCS)**, Volume 16 Issue 1Full text available: [pdf\(179.39 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Orca is a portable, object-based distributed shared memory (DSM) system. This article studies and evaluates the design choices made in the Orca system and compares Orca with other DSMs. The article gives a quantitative analysis of Orca's coherence protocol (based on write-updates with function shipping), the totally ordered group communication protocol,

the strategy for object placement, and the all-software, user-space architecture. Performance measurements for 10 parallel applications ill ...

Keywords: distributed shared memory, parallel processing, portability

4 A write update cache coherence protocol for min-based multiprocessors with accessibility-based split caches



M. S. Algudady, C. R. Das, M. J. Thazhuthaveetil
November 1990 **Proceedings of the 1990 ACM/IEEE conference on Supercomputing**

Full text available: [pdf\(941.95 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#)

In this paper, we present a cache coherence protocol for MIN-based multiprocessors with two distinct private caches: private-block caches containing information private to a processor, and shared-block caches containing data accessible by all processors. The protocol utilizes a coherence control bus (snooping) for connecting all shared-block cache controllers. Timing problems due to variable transit delays through the MIN are dealt with by introducing Transient states in this protocol. Assuming ...

5 SoftFLASH: analyzing the performance of clustered distributed virtual shared memory



Andrew Erlichson, Neal Nuckolls, Greg Chesson, John Hennessy
September 1996 **Proceedings of the seventh international conference on Architectural support for programming languages and operating systems**, Volume 31, 30 Issue 9 , 5

Full text available: [pdf\(1.29 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

One potentially attractive way to build large-scale shared-memory machines is to use small-scale to medium-scale shared-memory machines as clusters that are interconnected with an off-the-shelf network. To create a shared-memory programming environment across the clusters, it is possible to use a virtual shared-memory software layer. Because of the low latency and high bandwidth of the interconnect available within each cluster, there are clear advantages in making the clusters as large as possi ...

6 Design and performance of a coherent cache for parallel logic programming architectures



A. Goto, A. Matsumoto, E. Tick
April 1989 **ACM SIGARCH Computer Architecture News , Proceedings of the 16th annual international symposium on Computer architecture**, Volume 17 Issue 3

Full text available: [pdf\(1.17 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This paper describes the design and performance of a tightly-coupled shared-memory coherent cache optimized for the execution of parallel logic programming architectures. The cache utilizes a copy-back write-allocation protocol having five states and a hardware lock mechanism. Optimizations for logic programming are introduced in four software-controlled memory access commands: direct-write, exclusive-read, read-purge, and read-invalidate. In this paper we describe these operations and pres ...

7 Multi-level shared caching techniques for scalability in VMP-M/C



D. R. Cheriton, H. A. Goosen, P. D. Boyle
April 1989 **ACM SIGARCH Computer Architecture News , Proceedings of the 16th annual international symposium on Computer architecture**, Volume 17 Issue 3

Full text available: [pdf\(1.27 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The problem of building a scalable shared memory multiprocessor can be reduced to that of

building a scalable memory hierarchy, assuming interprocessor communication is handled by the memory system. In this paper, we describe the VMP-MC design, a distributed parallel multi-computer based on the VMP multiprocessor design, that is intended to provide a set of building blocks for configuring machines from one to several thousand processors. VMP-MC uses a memory hierarchy based on shared caches ...

8 STiNG: a CC-NUMA computer system for the commercial marketplace

Tom Lovett, Russell Clapp

May 1996 **ACM SIGARCH Computer Architecture News , Proceedings of the 23rd annual international symposium on Computer architecture**, Volume 24 Issue 2

Full text available:  pdf(1.30 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)



"STiNG" is a Cache Coherent Non-Uniform Memory Access (CC-NUMA) Multiprocessor designed and built by Sequent Computer Systems, Inc. It combines four processor Symmetric Multi-processor (SMP) nodes (called Quads), using a Scalable Coherent Interface (SCI) based coherent interconnect. The Quads are based on the Intel P6 processor and the external bus it defines. In addition to 4 P6 processors, each Quad may contain up to 4 GBytes of system memory, 2 Peripheral Component Interface (PCI) busses for ...

9 A taxonomy-based comparison of several distributed shared memory systems

Ming-Chit Tam, Jonathan M. Smith, David J. Farber

July 1990 **ACM SIGOPS Operating Systems Review**, Volume 24 Issue 3

Full text available:  pdf(1.96 MB)

Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)



Two possible modes of Input/Output (I/O) are "sequential" and "random-access", and there is an extremely strong conceptual link between I/O and communication. Sequential communication, typified in the I/O setting by magnetic tape, is typified in the communication setting by a **stream**, e.g., a UNIX¹ pipe. Random-access communication, typified in the I/O setting by a drum or disk device, is typified in the communication setting by **shared memory**. In this paper, we study and s ...

10 Cache coherence in large-scale shared-memory multiprocessors: issues and comparisons

David J. Lilja

September 1993 **ACM Computing Surveys (CSUR)**, Volume 25 Issue 3

Full text available:  pdf(3.12 MB)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)



11 Cache Memories

Alan Jay Smith

September 1982 **ACM Computing Surveys (CSUR)**, Volume 14 Issue 3

Full text available:  pdf(4.61 MB)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)



12 Dynamic self-invalidation: reducing coherence overhead in shared-memory multiprocessors

Alvin R. Lebeck, David A. Wood

May 1995 **ACM SIGARCH Computer Architecture News , Proceedings of the 22nd annual international symposium on Computer architecture**, Volume 23 Issue 2

Full text available:  pdf(1.37 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)



This paper introduces dynamic self-invalidation (DSI), a new technique for reducing cache

coherence overhead in shared-memory multiprocessors. DSI eliminates invalidation messages by having a processor automatically invalidate its local copy of a cache block before a conflicting access by another processor. Eliminating invalidation overhead is particularly important under sequential consistency, where the latency of invalidating outstanding copies can increase a program's critical path. DSI is ap ...

13 Missing the memory wall: the case for processor/memory integration

Ashley Sausbury, Fong Pong, Andreas Nowatzky

May 1996 **ACM SIGARCH Computer Architecture News, Proceedings of the 23rd annual international symposium on Computer architecture**, Volume 24 Issue 2

Full text available:  [pdf\(1.45 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)



Current high performance computer systems use complex, large superscalar CPUs that interface to the main memory through a hierarchy of caches and interconnect systems. These CPU-centric designs invest a lot of power and chip area to bridge the widening gap between CPU and main memory speeds. Yet, many large applications do not operate well on these systems and are limited by the memory subsystem performance. This paper argues for an integrated system approach that uses less-powerful CPUs that are ...

14 Micro benchmark analysis of the KSR1

R. H. Saavedra, R. S. Gaines, M. J. Carlton

December 1993 **Proceedings of the 1993 ACM/IEEE conference on Supercomputing**

Full text available:  [pdf\(1.77 MB\)](#)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)



15 Cache coherence protocols: evaluation using a multiprocessor simulation model

James Archibald, Jean-Loup Baer

September 1986 **ACM Transactions on Computer Systems (TOCS)**, Volume 4 Issue 4

Full text available:  [pdf\(1.79 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)



Using simulation, we examine the efficiency of several distributed, hardware-based solutions to the cache coherence problem in shared-bus multiprocessors. For each of the approaches, the associated protocol is outlined. The simulation model is described, and results from that model are presented. The magnitude of the potential performance difference between the various approaches indicates that the choice of coherence solution is very important in the design of an efficient shared-bus multi ...

16 Algorithms for scalable synchronization on shared-memory multiprocessors

John M. Mellor-Crummey, Michael L. Scott

February 1991 **ACM Transactions on Computer Systems (TOCS)**, Volume 9 Issue 1

Full text available:  [pdf\(3.07 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)



Busy-wait techniques are heavily used for mutual exclusion and barrier synchronization in shared-memory parallel programs. Unfortunately, typical implementations of busy-waiting tend to produce large amounts of memory and interconnect contention, introducing performance bottlenecks that become markedly more pronounced as applications scale. We argue that this problem is not fundamental, and that one can in fact construct busy-wait synchronization algorithms that induce no memory or interc ...

17 Distributed file systems: concepts and examples

Eliezer Levy, Abraham Silberschatz

December 1990 **ACM Computing Surveys (CSUR)**, Volume 22 Issue 4



Full text available:  pdf(5.33 MB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

The purpose of a distributed file system (DFS) is to allow users of physically distributed computers to share data and storage resources by using a common file system. A typical configuration for a DFS is a collection of workstations and mainframes connected by a local area network (LAN). A DFS is implemented as part of the operating system of each of the connected computers. This paper establishes a viewpoint that emphasizes the dispersed structure and decentralization of both data and con ...

18 Architecture: Leveraging cache coherence in active memory systems 

Daehyun Kim, Mainak Chaudhuri, Mark Heinrich

June 2002 **Proceedings of the 16th international conference on Supercomputing**Full text available:  pdf(217.27 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Active memory systems help processors overcome the memory wall when applications exhibit poor cache behavior. They consist of either active memory elements that perform data parallel computations in the memory system itself, or an active memory controller that supports address re-mapping techniques that improve data locality. Both active memory approaches create coherence problems---even on uniprocessor systems---since there are either additional processors operating on the data directly, or the ...

Keywords: active memory, address re-mapping, cache coherence**19 The VMP multiprocessor: initial experience, refinements, and performance evaluation** 

D. R. Cheriton, A. Gupta, P. D. Boyle, H. A. Goosen

May 1988 **ACM SIGARCH Computer Architecture News , Proceedings of the 15th Annual International Symposium on Computer architecture**, Volume 16 Issue 2Full text available:  pdf(1.73 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

VMP is an experimental multiprocessor being developed at Stanford University, suitable for high-performance workstations and server machines. Its primary novelty lies in the use of software management of the per-processor caches and the design decisions in the cache and bus that make this approach feasible. The design and some uniprocessor trace-driven simulations indicating its performance have been reported previously. In this paper, we present our initial experience with the V ...

20 Exploiting high-level coherence information to optimize distributed shared state 

DeQing Chen, Chunqiang Tang, Brandon Sanders, Sandhya Dwarkadas, Michael L. Scott

June 2003 **ACM SIGPLAN Notices , Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming**, Volume 38 Issue 10Full text available:  pdf(841.96 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

InterWeave is a distributed middleware system that supports the sharing of strongly typed, pointer-rich data structures across a wide variety of hardware architectures, operating systems, and programming languages. As a complement to RPC/RMI, InterWeave facilitates the rapid development of maintainable code by allowing processes to access shared data using ordinary reads and writes. Internally, InterWeave employs a variety of aggressive optimizations to obtain significant performance improvements ...

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2005 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads: [Adobe Acrobat](#) [QuickTime](#) [Windows Media Player](#) [Real Player](#)